[10191/3300]

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants	:	Michael BEUTEN et al.
For	:	METHOD FOR DYNAMIC MEMORY MANAGEMENT
Application Serial No.	:	10/633,113
Filed	:	August 1, 2003
Examiner	:	Mardochee CHERY
Group Art Unit	:	2188
Confirmation No.	:	3639

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to: Mail Stop Appeal Brief-Patents, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on:

Date: September 4, 2007 Reg. No. 36,197

Signature: Jong H. Lee

APPELLANTS' APPEAL BRIEF
UNDER 37 C.F.R. § 41.37

S I R :

Applicants filed a Notice of Appeal dated April 27, 2007 (received at the PTO on May 1, 2007), appealing from the Final Office Action dated November 2, 2006, in which claims 1-10 and 12 of the above-identified application were finally rejected. This Brief is submitted by Applicant in support of their appeal.

I. REAL PARTY IN INTEREST

The real party in interest in the present appeal is Robert Bosch GmbH of Stuttgart, Germany. Robert Bosch GmbH is the assignee of the entire right, title, and interest in the present application.

II. RELATED APPEALS AND INTERFERENCES

No appeal or interference which will directly affect, or be directly affected by, or have a bearing on, the Board's decision in the pending appeal is known to exist to the undersigned attorney or is believed by the undersigned attorney to be known to exist to Applicants.

III. STATUS OF CLAIMS

Claims 1-10 and 12 are pending in the present application, and claims 1-10 and 12 are being appealed. Among the appealed claims, claims 1, 6, 8 and 12 are independent; claims 2-5 ultimately depend on claim 1; claim 7 ultimately depend on claim 6; claims 9-10 ultimately depend on claim 8. Claim 11 was canceled in the Amendment dated February 10, 2006.

IV. STATUS OF AMENDMENTS

No amendment has been made subsequent to the final Office Action mailed on November 2, 2006.

V. SUMMARY OF CLAIMED SUBJECT MATTER

With respect to independent claim 1, the present invention provides a method for providing dynamic memory management of a memory device (Fig. 1, element 14), the method including:

providing a first memory block (Fig. 2, element 30; Fig. 4, element 70) in the memory device; (specification, p. 5, l. 5-7);

storing a startup program in the first memory block; (p. 5, l. 5-7; p. 6, l. 15-16 and 23-24);

providing additional memory blocks (Fig. 2, elements 32-36; Fig. 4, elements 72-76); (p. 5, l. 12-25; p. 6, l. 16-17 and 25-30);

connecting the first memory block and the additional memory blocks by a chained list (Fig. 1, arrows 20; Fig. 3, arrows 42, 44); (p. 4, l. 25-26; p. 5, l. 27-30);

wherein the memory device is checked before the chained list is executed and the startup program obtains data for a check from the additional memory blocks (p. 2, l. 25-27 and 34; p. 6, l. 1-3 and p. 6, l. 34 – p. 7, l. 2).

With respect to independent claim 6, the present invention provides a memory device (Fig. 1, element 14) including:

a first memory block (Fig. 2, element 30; Fig. 4, element 70) to store a startup program; (p. 5, l. 5-7; p. 6, l. 15-16 and 23-24); and

additional memory blocks (Fig. 2, elements 32-36; Fig. 4, elements 72-76) to store data for a check; (p. 4, l. 32-33; p. 6, l. 16-17 and 25-30);

wherein the first memory block and the additional memory blocks are connected by a chained list (Fig. 1, arrows 20; Fig. 3, arrows 42, 44; p. 4, l. 25-26; p. 5, l. 27-30), and wherein the memory device is checked before the chained list is executed and the startup program obtains data for a check from the additional memory blocks (p. 2, l. 25-27 and 34; p. 6, l. 1-3 and p. 6, l. 34 – p. 7, l. 2).

With respect to independent claim 8, the present invention provides a system including:

a computing unit (Fig. 1, element 12);

a memory device (Fig. 1, element 14) including a first memory block (Fig. 2, element 30; Fig. 4, element 70) to store a startup program; (p. 5, l. 5-7; p. 6, l. 15-16 and 23-24); and

additional memory blocks (Fig. 2, elements 32-36; Fig. 4, elements 72-76) to store data for a check; (p. 4, l. 32-33; p. 6, l. 16-17 and 25-30);

wherein the first memory block and the additional memory blocks are connected

by a chained list (Fig. 1, arrows 20; Fig. 3, arrows 42, 44; p. 4, l. 25-26; p. 5, l. 27-30), and wherein the memory device is checked before the chained list is executed and the startup program obtains data for a check from the additional memory blocks (p. 2, l. 25-27 and 34; p. 6, l. 1-3 and p. 6, l. 34 – p. 7, l. 2).

With respect to independent claim 12, the present invention provides a computer-readable storage medium including program code (p. 3, l. 29-37) for providing dynamic memory management of a memory device (Fig. 1, element 14), the program code being executable in a computing arrangement to perform the following:

providing a first memory block (Fig. 2, element 30; Fig. 4, element 70) in the memory device; (p. 5, l. 5-7);

storing a startup program in the first memory block; (p. 5, l. 5-7; p. 6, l. 15-16 and 23-24);

providing additional memory blocks; (Fig. 2, elements 32-36; Fig. 4, elements 72-76); (p. 5, l. 12-25; p. 6, l. 16-17 and 25-30); and

connecting the first memory block and the additional memory blocks by a chained list; (Fig. 1, arrows 20; Fig. 3, arrows 42, 44); (p. 4, l. 25-26; p. 5, l. 27-30);

wherein the memory device is checked before the chained list is executed and the startup program obtains data for a check from the additional memory blocks (p. 2, l. 25-27 and 34; p. 6, l. 1-3 and p. 6, l. 34 – p. 7, l. 2).

VI. GROUND OF REJECTION TO BE REVIEWED ON APPEAL

The following ground of rejection is presented for review on appeal in this case:

(A) Whether claims 1-10 and 12 are unpatentable under 35 U.S.C. § 103(a) over U.S. Patent No. 6,088,777 ("Sorber") in view of U.S. Patent No. 6,192,457 ("Porterfield") in view of U.S. Patent No. 6,141,756 ("Bright").

VII. ARGUMENTS

Claims 1-10 and 12 were rejected under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 6,088,777 ("Sorber") in view of U.S. Patent No. 6,192,457 ("Porterfield")

in view of U.S. Patent No. 6,141,756 ("Bright"). Applicants respectfully submit that claims 1-10 and 12 are not rendered obvious by Sorber, Porterfield and Bright, for at least the reasons set forth below.

In order for a claim to be rejected for obviousness under 35 U.S.C. § 103(a), the prior art teach or suggest each element of the claim. See Northern Telecom, Inc. v. Datapoint Corp., 908 F.2d 931, 934 (Fed. Cir. 1990), cert. denied, 111 S. Ct. 296 (1990); In re Bond, 910 F.2d 831, 834 (Fed. Cir. 1990). To establish a *prima facie* case of obviousness, the Examiner must show, *inter alia*, that there is some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify or combine the references, and that, when so modified or combined, the prior art teaches or suggests all of the claim limitations. M.P.E.P. §2143. In addition, as clearly indicated by the Supreme Court, it is "important to identify a reason that would have prompted a person of ordinary skill in the relevant field to combine the [prior art] elements" in the manner claimed. See KSR Int'l Co. v. Teleflex, Inc., 127 S. Ct. 1727 (2007). To the extent that the Examiner may be relying on the doctrine of inherent disclosure for the obviousness rejection, the Examiner must provide a "basis in fact and/or technical reasoning to reasonably support the determination that the allegedly inherent characteristics necessarily flow from the teachings of the applied art." (See M.P.E.P. § 2112; emphasis in original; see also Ex parte Levy, 17 U.S.P.Q.2d 1461, 1464 (Bd. Pat. App. & Inter. 1990)).

Independent claim 1 recites, in relevant parts, "providing **a first memory block in the memory device**; storing a **startup program in the first memory block**; providing additional memory blocks; and **connecting the first memory block and the additional memory blocks by a chained list**; wherein the memory device is checked before the chained list is executed and the **startup program obtains data for a check from the additional memory blocks**." Independent claims 6, 8 and 12 recite substantially similar features as the above-recited features of claim 1.

In support of the rejection, the Examiner contends that Sorber discloses the claimed features of "storing **a startup program** in the first memory block [of the memory device] (col. 7, par. 2)." However, Sorber explicitly indicates that the startup program is contained in the program memory 20, which is separately located from the data memory 22 (see, e.g., Fig. 2, which shows program memory 20 connected by an external communications bus 18 to a separately located data memory 22, as described in col. 7, l. 1-14). To the extent the Examiner contends in the Advisory Action that col. 7, l. 1-14 of Sorber does not teach that the program memory 20 is separate from the data memory 22, the Examiner's interpretation simply does not make any sense in light of the facts that the program memory 20 and the data memory 22 are shown to be separately located in Fig. 2, and therefore memories 20 and 22 need to be connected by an external communications bus 18. To the extent the Examiner may be contending that the Examiner is entitled to assume that memories 20 and 22 of Sorber are not separate because there is no explicit statement in Sorber that memories 20 and 22 are separate, Applicants note that it is the Examiner's burden to initially establish a prima facie case of obviousness with respect to each element, which the Examiner has failed to establish in this case.

In addition, to the extent the Examiner cites col. 16, l. 54-57 of Sorber as teaching the claimed feature of "connecting the first memory block and the additional memory blocks by a chained list," Applicants note that col. 16, l. 54-57 merely indicates that "each available memory block is linked to a next, available memory block in the same class and a first available memory block in the same class is indicated by a pointer," which statement has nothing to do with "storing a **startup program in the first memory block** . . . and **connecting the first memory block [storing the startup program] and the additional memory blocks by a chained list**." In other words, "each available memory block" recited in col. 16, l. 54-57 clearly does not refer to the memory block containing the startup program; instead, "each available memory block is linked to a next, available memory block in the same class and a first available memory block in the same class is indicated by a pointer" recited in claim 34 of Sorber refers to the description in col. 8, l. 55-61, i.e., "If the memory block is a part of a message, it may be linked to other blocks in a list, (i.e., string of linked blocks), before the pointer to the first block

in the message string is given to the next process. The memory manager 26 moves the class 1 available memory block pointer to the next memory block in that class." Accordingly, the cited statement of Sorber that "each available memory block is linked to a next, available memory block in the same class and a first available memory block in the same class is indicated by a pointer" has absolutely nothing to do with linking of the memory block containing the startup program, and there is absolutely no teaching or suggestion in Sorber that the memory block containing the startup program (which is clearly in program memory 20) is connected to additional memory blocks by a chained list, let alone connected to the "available memory block" recited in col. 16, l. 54-57.

To the extent the Examiner contends in the Advisory Action that "Applicants' argument appear as though the claim recites 'a memory block referring to the memory block containing the startup program,' . . . [which] limitation is not recited in the claim," and that "the features upon which applicant relies (i.e., linking of the memory block containing the startup program and memory block containing the startup program connected to additional memory blocks by a chained list) are not recited in the rejected claim(s)," Applicants note that these contentions are clearly wrong: the claimed language requires "**storing a startup program in the first memory block**" and "**connecting the first memory block and the additional memory blocks by a chained list,**" which necessarily mean that the "first memory block" recited in the phrase "**connecting the first memory block and the additional memory blocks**" stores "a **startup program**." Accordingly, there is absolutely no basis for the Examiner to contend that Applicants are relying on features which are not recited in the rejected claims.

Independent of the above, regarding the feature that "the **memory device is checked** before the chained list is executed and the **startup program obtains data for a check** from the **additional memory blocks**," the Examiner contends that "Porterfield discloses the startup program obtains data for a check from the additional memory blocks (col. 4, lines 3-13) to initialize the computer system upon being turned on (col. 4, lines 11-13)." (Final Office Action, p. 4). However, col. 4, lines 3-13 of Porterfield has nothing to do with the startup

program obtaining **data for a check of the memory device**, which data are obtained from the additional memory blocks; instead, this cited section of Porterfield merely discloses a system allocation table specifying system addresses for the various components of the computer system, and that the "addresses allocated for each computer device in the system address allocation table typically will be set by the Basic Input-Output System (BIOS) software when the computer system 50 is initialized upon being turned ON." Accordingly, col. 4, lines 3-13 of Porterfield has nothing to do with a check of the memory device, let alone anything to do with **the startup program obtaining data for a check of the memory device from the additional memory blocks**.

To the extent the Examiner contends in the Advisory Action that "BIOS is the set of essential software routines that tests hardware at startup," and thus "the BIOS in the system of Porterfield obtains data from a first portion of memory which is then tested at startup," (Advisory Action, paragraph 4), Applicants note that this assertion in the Advisory Action contradicts, and does not support, the Examiner's earlier assertion in the Final Office Action that "the startup program obtains data for a check from the additional memory blocks." Even if one assumes for the sake of argument that BIOS tests hardware at startup, this assumption simply does not provide any teaching or suggestion regarding **where the startup program (which is stored in the first memory block) obtains data for the check**, let alone provide any suggestion that **the startup program stored in the first memory block obtains data for a check of the memory device from the additional memory blocks**.

Independent of the above, regarding the feature that "**the memory device** [containing the startup program and the additional memory blocks] **is checked before the chained list is executed**," the Examiner contends that Bright discloses this feature in col. 3, l. 15-27. However, in contrast to the Examiner's contention, there is absolutely no suggestion in Bright regarding any memory device containing the startup program and the additional memory blocks is checked, let alone that any such memory device is checked before any chained list is

executed; instead, the cited section of Bright merely indicates use of encryption/decryption in downloading a program from an external device 103 by a separate processor 101.

To the extent the Examiner contends in the Advisory Action that "one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references," Applicants note that the Applicants' discussions of the applied references simply mirror the Examiner's discussions of the applied references, i.e., the Examiner discusses the teachings of each individual references as a part of the discussion of the alleged overall teachings of the applied references, and Applicants are simply explaining the deficiencies in the Examiner's discussions of each individual references. If the Examiner can discuss the overall teachings of the applied reference without discussing the teachings of each individual references, then Applicants would be happy to respond in kind.

Independent of the above, to the extent the Examiner contends in paragraph 5 of the Advisory Action that "Bright unequivocally discloses 'the program stored in the external device has added authenticity information, such as a checksum, hash and so forth, which authenticity information must be authenticated by the processor before the program may be executed; col. 3, ll. 15-27,' where it is readily apparent that an executable program includes chained lists that upon execution of the program are also being executed," Applicants submit that this contention is a complete nonsense: even if "Bright unequivocally discloses 'the program stored in the external device has added authenticity information, such as a checksum, hash and so forth, which authenticity information must be authenticated by the processor before the program may be executed," this assumption has absolutely no logical connection to the assertion that "**an executable program includes chained lists** that upon execution of the program are also being executed" as contended by the Examiner, let alone provide any suggestion that a **memory device containing the startup program and the additional memory blocks** is checked, or that any such memory device is checked **before any chained list is executed**.

Not only do the overall teachings of Sorber, Porterfield and Bright fail to suggest all of the features of independent claims 1, 6, 8 and 12, but there is no logical reason why any person of ordinary skill in the art would be motivated to make the modifications asserted by the Examiner, particularly when one considers the completely different technologies involved in the applied references. For example, the teachings of Bright involve the use of encryption/decryption in downloading a program from an external device 103 by a separate processor 101, while the teachings of Porterfield involve implementing a graphics address remapping table as a virtual register in system memory, and the teachings of Sorber involve a memory management scheme for dynamic memory allocation corresponding to various data sizes. There is simply no logical reason why one of ordinary skill in the art would look to these completely unrelated teachings of Sorber, Porterfield and Bright, let alone any reason why one of ordinary skill in the art would specifically combine the selected teachings of these applied reference in the manner asserted by the Examiner. To the extent the Examiner contends in the Advisory Action that Sorber, Porterfield and Bright involve the same technology because each of these references involve a memory-related operation, the Examiner is completely ignoring the detailed teachings of the three references, which are clearly unrelated.

For at least the foregoing reasons, the overall teachings of Sorber, Porterfield and Bright cannot possibly render independent claims 1, 6, 8 and 12, as well as their dependent claims 2-5, 7, 9 and 10, obvious.

VIII. CONCLUSION

For the foregoing reasons, it is respectfully submitted that the final rejections of claims 1-10 and 12 should be reversed.

Claims Appendix, Evidence Appendix and Related Proceedings Appendix sections are found in the attached pages.

Respectfully submitted,

KENYON & KENYON LLP

Dated: September 4, 2007

By: JONG LEE for Gerard Messina

Gerard A. Messina
Reg. No. 35,952
One Broadway
New York, NY 10004
(212) 425-7200
CUSTOMER NO. 26646

APPENDIX TO APPELLANTS' APPEAL BRIEF
UNDER 37 C.F.R. § 41.37

CLAIMS APPENDIX

The claims involved in this appeal, claims 1-10 and 12, in their current form after entry of all amendments presented during the course of prosecution, are set forth below:

1. A method for providing dynamic memory management of a memory device, the method comprising:

providing a first memory block in the memory device;

storing a startup program in the first memory block;

providing additional memory blocks; and

connecting the first memory block and the additional memory blocks by a chained list;

wherein the memory device is checked before the chained list is executed and the startup program obtains data for a check from the additional memory blocks.

2. The method of claim 1, wherein the checking is performed using an addition checksum.

3. The method of claim 1, wherein the checking is performed by a cyclic block backup.

4. The method of claim 1, wherein the checking is performed at a time of booting a system that includes the first memory block and the additional memory blocks.

5. The method of claim 1, wherein the checking is performed in the background during operation of a system that includes the first memory block and the additional memory blocks.

6. A memory device, comprising:

a first memory block to store a startup program; and

additional memory blocks to store data for a check;

wherein the first memory block and the additional memory blocks are connected by a chained list wherein the memory device is checked before the chained list is executed and the startup program obtains data for a check from the additional memory blocks.

7. The memory device of claim 6, wherein each of the additional memory blocks includes an information area that stores information on the memory block itself and a checking area that stores information for performing the check.

8. A system, comprising:

a computing unit;

a memory device including a first memory block to store a startup program; and

additional memory blocks to store data for a check;

wherein the first memory block and the additional memory blocks are connected by a chained list, wherein the memory device is checked before the chained list is executed and the startup program obtains data for a check from the additional memory blocks.

9. The system of claim 8, wherein the memory device includes a non-volatile memory module.

10. The system of claim 8, wherein the computing unit includes an embedded microcontroller.

12. A computer-readable storage medium including program code for providing dynamic memory management of a memory device, the program code being executable in a computing arrangement to perform the following:

providing a first memory block in the memory device;

storing a startup program in the first memory block;

providing additional memory blocks; and

connecting the first memory block and the additional memory blocks by a chained list;

wherein the memory device is checked before the chained list is executed and the startup program obtains data for a check from the additional memory blocks.

EVIDENCE APPENDIX

In the present application, there has been no evidence submitted pursuant to 37 C.F.R. §§ 1.130, 1.131 or 1.132, or other evidence entered by the Examiner and relied upon by Appellants in the present appeal.

RELATED PROCEEDINGS APPENDIX

No appeal or interference which will directly affect, or be directly affected by, or have a bearing on, the Board's decision in the pending appeal is known to exist.